Log System Based on Software Testing System Design And Implementation

Yan Liu[1, a], Dahai Jin[1, b] and Dalin Zhang[2, c]

[1] Institute of Network Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China;

[2] Institute of Network Technology, Beijing Jiaotong University, Beijing, 100044, China.

[a]13161892342@163.com, [b]jindh@bupt.edu.cn, [c]dalin@bjtu.edu.cn .

Keywords: Log system, software testing system, testing system.

Abstract. Defect testing system is developed in recent years, which is a source code static analysis technology, mainly used in automatic or semi-automatic software defect detection and prevention, this kind of defect testing tools due to the characteristics of high efficiency and easy to use, has been widely used in high availability software testing. And in the process of software development and system running, and the log management is a very important part. Detect testing system analysis is a very complex process that requires a defect testing logging system output information of system operation process, to provide users with sufficient information system operation. At the same time, output program operation exception information, it is convenient for developers quickly locating and solving problems.

1. Introduction

Log system is an indispensable trace debugging tools, for a long time, logging system as a kind of application services for debugging. The collapse of the program status records, data recovery has very realistic significance. Debug a Java program on the console environment, in order to check the program running state, often console or text file output a paragraph of text, this way is simpler, but with a lot of print statements in your code, code redundancy and not beautiful[1]. The random output information is not controllable, debugging information is not standard, it is difficult to remove, may bring the background monitoring, troubleshooting and error recovery considerable resistance.

In the process of software development and system operation, log management is a very important part. Statistics show that in a system log output in the amount of code that account for about 4% of the total amount of code. Early practice is embedded in a lot of print statements in your code, the print statements can be output to the console or file, then development to construct a log operation class to encapsulate such operations, rather than making a series of print statements is the main body of code. In this paper, based on the famous Apache Log4j, an open source project based on a defect detection system log system, achieve the function of defect detection system log management.

2. Research of existing log technology

2.1 A complete log system framework

A complete log system framework should typically include the following basic characteristics:

(1) Log level classification: log by some standards are divided into different levels, the grading after logging, log filter can be used under the same classification.

(2) Support multithreading: In Java system, logging system is usually used in a multithreaded environment, therefore, as a kind of system resources, logging system should guarantee is thread-safe.

(3) Log classification: Log system should have their own classification of the system output, which facilitate when debugging to query system of different modules, so as to quickly locate to events code.

(4) Different recording media: Different projects tend to record medium requirements for logging system is different, therefore, the log system must provide the necessary development interface, ensure can easily change the recording medium.

(5) Stability: The logging system is must maintain a high level of stability, the log system internal error led to the collapse of the main business code, it is cannot happen.

2.2 Introduction of basic log system

As a conclusion, we can't check the system output for a system, or keep the output system information, a good logging system is quite necessary. In the Java world, the following three logging framework is good:

(1) Log4j

Log4j is one of the earliest Java logging framework, has been initiated by the Apache foundation, provide flexible and powerful logging mechanism.

(2) JDK1.4LoggingFramework

After the Log4j, JDK standards committee will absorb the basic idea of Log4j to JDK, issued the first log frame in JDK1.4 interface, and provides a simple implementation.

(3) Commons Logging Framework

The framework is also the Apache foundation projects, its occur mainly in order to make a Java project in Log4j and JDK1.4 Logging Framework optional to switch on the use, so the framework provides a unified call interface and configuration method.

3. Introduction of software testing system

Software testing system based on the analysis of the source code and processing, detect the defects of the source code, and finally generate a defect inspection report to the user, its specific analysis process is as follows:

(1) Construct the abstract syntax tree: scan the source code, after pretreatment, lexical analysis, syntax analysis, produce and process corresponds to the abstract syntax tree[2]. The abstract syntax tree is the basis of the subsequent analysis, is the preliminary abstract source program.

(2) Generate the control flow graph: control flow graph reflects the control structure of the program. Program control flow graph and is corresponding to the syntax tree, every node on the control flow chart corresponding to the statements of syntax tree node. System can access syntax tree from the control flow graph. In the same way, from the statement of the syntax tree node can access to control flow graph of the corresponding node.

(3) Generate the symbol table: from the abstract syntax tree to generation symbol table, the symbol table are used to record the type and scope of the identifier and the binding information. The symbol table will be mapping identifier and its type and location[3].

(4) Generate the using - definition chain and the definition - using chain: data flow can be used to calculate the variable's using - definition chain and the definition - using chain. For each use of variable x, the use of variable x's using - definition chain saved all the information which is likely to reach the current use of the definition of x. Definition – using chain is to save the current definition lists of all the possible use[4].

(5) Generate call graph: function call relationship can be represented with a directed graph, the nodes in the graph to represent the function, side on behalf of the call. If the function f () calls the function g (), then in the function call graph corresponding the existence of the f node to g node. Call graph is often used in the process of program analysis[5], for example: if you want to calculation function return value scope, then each function suitable calculation sequence is a figure of a topological sort function call.

(6) Defect pattern state machine resolution: parsing defect mode in the database defect description file, generate the defect detection of internal data structure of the state machine.

(7) Variable interval analysis: static analysis program is the range of the value of the variable in the analysis of improving testing precision and efficiency is an important basis. By extending the

range of abstract interpretation theory, numerical interval arithmetic and Boolean variable[6], support reference variables and array variable interval operation, and get all the variables of program scope.

(8) Automatic test and analysis: according to the test configuration, the module for the application under test to produce different defect pattern state machine instance and defect pattern matching, matching process is calculate according to the state conditions along the flow of control state change of each state machine instance. If on the control flow chart of a particular node error then to the checkpoint report output a defect record.

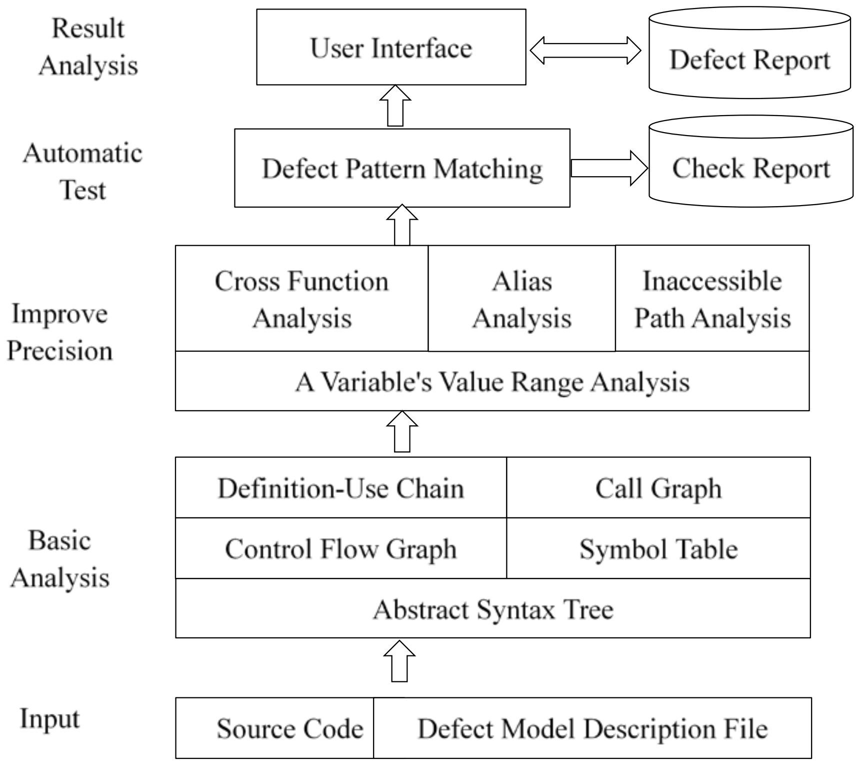

Fig. 1 Defect testing system frame diagram

4. Existing problems of logging framework

Now, software defect testing system log system can meet the demand of basic logging requirement, the user can be obtained by setting different levels of logging information, can view the system log information, but at present there are still some problems.

(1) Log information is incomplete

Software defect testing system in the analysis of the source code process is relatively complex, but the current logging system to print out the log information is rough, can only get the general steps of analysis and the analysis time, and many detailed analysis process is not printed, so cause the log information is not complete, the programmer can't according to the log information to judge the analysis process of the system.

(2) Log information is confusion

In the process of software defect testing system based on the analysis of the source code, may encounter some current software defect testing system can't handle the source code in some situation, such as user defined syntax and grammar mistakes, state machine, created problems, etc., software defect testing system can't identify and quote us grammatical errors. And by running the software defect testing system analysis of the source code to get the log information, basically cannot as a programmer debugging reference information, and in many cases, programmers still need manual debugging process.

(3)Cannot get consistent results

When the software defect testing system based on the analysis of some complex source code, there might be two analysis results are inconsistent, but through the current running results, can't analysis two analysis results are inconsistent, so we need to log system perfect, get more detailed log information, so that through the analysis of log information, find out two software defect testing system test results are inconsistent.

5. The design of logging system

5.1 Log system architecture

Logging system framework can be divided into the logging and log output module of two parts. Logging module is responsible for creating and managing the logger, each logger object is responsible for receiving log objects which is recorded by the log system, and log object is divided into different levels. Logger objects first obtain all need to record the log, and synchronously the log assigned to log output module. The log output module is responsible for the creation and management of log output, as well as the log output. Log system allows for a number of different log output, log output device is responsible for logging into the storage medium. Logging system structure is as follows.

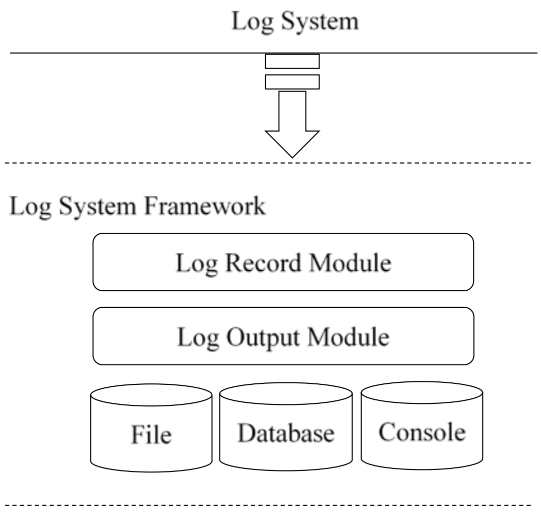

Fig. 2 Logging system structure

In fig.3 shows the system architecture, the logger is the user interface of the logging system framework, the programmer can log information through the interface. For classifying the log, system design allows multiple Logger object, each Logger is responsible for a class of log records, logger class implements the management of the object itself at the same time. Logger Level class defines the level of logging system, these levels can be used when the client to create and send log. Logger object is received the client to create and send the log message, at the same time to pack the log messages into the log object Log Item logging system internal use. Log object in addition to the messages sent by the sender, also the packing, such as the sender name of the class, sending events, sending the method name, and so on. These additional messages for system tracking and debugging are very valuable. Packaged Log item was eventually sent to the follower, the follower is responsible for the log information of writing the final medium, the type and number of follower are not fixed. Manage all follower by Appender Manager, usually through the configuration file can be convenient to expand the multiple output device.

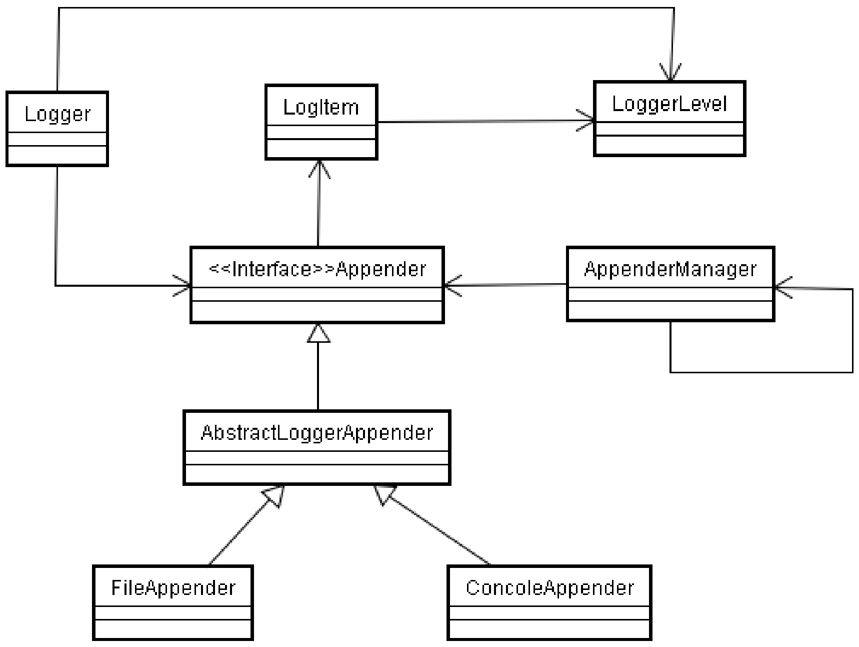

Fig. 3 Log system architecture diagram

5.2 The design of the log output section

Log output part of the design has certain difficulty, in the log system designed in this paper, the support of the log output, multithreading, log system problems such as extensibility, efficiency of logging system to manage log output section.

1 Inheritance of log output structure

In the output section of the log used the layer structure, which defines an abstract logger appender. Abstract logger appender defines a series of log filtering method, and the method of the output to a storage medium is an abstract method, by a subclass implementation. In the system default implementation the console output device and file output device, including the realization of the console output device is simple.

2 Internal implementation of file output device

In the realization of the logging section, and did not consider multithreading, the problem such as high efficiency, so the file output device must consider these problems. Inside the file output device using vector defined a thread-safe cache, all through the log file output part assigned to implement the logs are placed directly into the cache. Inside the file output device at the same time defines a worker thread, is responsible for the regular will cache the contents of the save to file, in the process of saving at the same time can be a backup log files, etc. Due to adopt the structure of the cache, it is clear that the log is no longer a client calls a synchronous invocation, which will no longer need to return until after file operations, improve the speed of the system call. File output device of the schematic diagram is shown in fig 4.

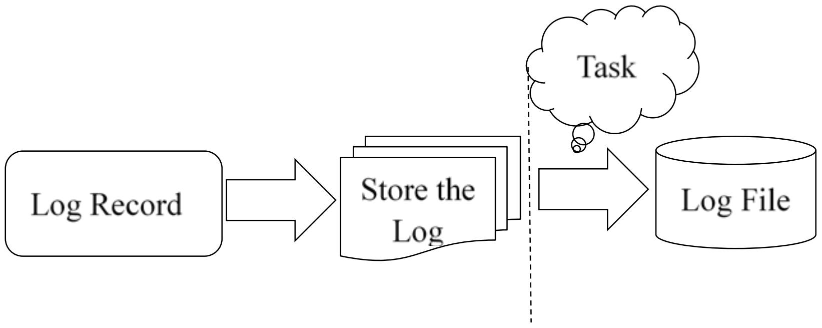

Fig. 4 Log system architecture diagram

6. The realization of the log system

6.1 Tracking function

Software defect testing system of logging system main function should have the function of tracking analysis process, abnormal positioning function and management function.

Through the log system able to track software defects testing system of the specific analysis. By controlling can get different levels of logging, through detailed log information, you can see detailed analysis software defect testing system detection process, track processing source code analysis of each process.

Software defect testing system analysis source mainly include: the process of pretreatment process, document analysis, function analysis and instance analysis. Through the detailed log information log system, able to track every step of the process in detail.

Pretreatment process is divided into: generate the abstract syntax tree, generate the symbol table and complete the global function analysis.

File analysis process is divided into: generated file analysis, topological sort order analysis.

Function analysis process is divided into: create an abstract syntax tree, generate the symbol table, global function analysis, generating function call diagram, generate the control flow graph and computing definition - using chain, etc.

The example analysis process is divided into: quote us found the failure mode.

6.2 Abnormal positioning function

Exception is refers to the unexpected conditions, such as file can't find, network connection fails, illegal parameters, etc. Exception occurs during the operation of the program, it can interfere with the normal process of instruction. Similarly, in the process of software defect testing system of testing, also find some unexpected exception, raises some unknown error, we hope to throw these exceptions, able to quickly locate through the log information to appear abnormal.

6.3 Management functions

(1) Switch management

By adding various control switch, to control the effect of log information.By logging system switch function, we can detailed grain size, convenient and effective control log information for temporarily don't pay attention to the log information can choose to switch, so you can get the log information of particle size.

(2) Different level log points file record

In software defect testing system log system, different level of logging information to record in a different file. In software defect testing system log system, it is recommended to use four kinds of log level: DEBUG, INFO, WARN, and ERROR, the log level four types of log information output to the four log file.

7. Conclusion and future work

In this paper, the author design and implement a log system. It bring some benefits to our later work.

(1) Assistant development

The log system can help developers to develop, and developers can easily browse through log system software defect testing system of general analysis steps. Programmers don't have to every time debugging, and debugging the efficiency is low. Through the analysis of view software defect testing system logs, can quickly understand the working process of the system, to help developers understand the system in order to develop rapidly.

At the same time, when in the process of system analysis, abnormal information, by looking at the log information will also be able to quickly find the problem and help to quickly solve problems.

(2) Log information standard, convenient view

Logging system log information format is unified, can easily through the different parameter control of switch and view the log information of different granularity. It obtained by script after processing, can also be one kind of logging information, convenient viewing.

(3) Configure the log information is simple and effective

Users can through the one ways of simple and effective control of logging information. The Operation of the first open the software defect testing system interface, selecting log option switch Settings for configuration. Through some simple parameters such as configuration of "true" and "false" information will be able to control a lot of log switch.

In the future, we need to continue to improve the log system, so as to get more detailed log information.

[1]Supported by the National Natural Science Foundation of China (Grant No. 61502029)

References

[1]. Quinlan Daniel J, Vuduc Richard W, Misherghi Ghassan. Techniques for specifying bug patterns. In Proceedings of the 2007 ACM workshop on Parallel and distributed systems:testing and debugging, London, 2007:27-35.

[2]. Cousot P. Abstract interpretation based formal methods and future challenges, invited paper. In R. Wilhelm, editor, Informatics-10 Years Back, 10 Years Ahead, Volumn 2000 of LNCS, Springer-Verlag, 2000:138-156

[3]. King J C.Symbolic execution and testing. Comm. Of the ACM, 1976,19:385-394

[4]. Nelson G. Oppen D.Simplification by coorperating decision procedures. ACM TOPLAS, 1979,1(2):245-257

[5]. Dwyer Matthew B, Hatcliff John, Robby, et al.Formal Software Analysis Emerging Trends in Software Model Checking. In Proceedings of International Conference on Software Engineering, 2007:120-136.

[6]. Hallem S. Chelf B, Xie Y, et al. A system and language for building system-specific static analyses. In Proceedings of the 2002 ACM SIGPLAN Conference on Programming Language Design and Implemention, 2002:69-82.